RSID 1-289
5244-0107-2

TITLE OF THE INVENTION

NETWORK FAX MACHINE USING A WEB PAGE AS A USER INTERFACE

BACKGROUND OF THE INVENTION

Field of the Invention:

The present invention relates generally to a network fax machine and more specifically to an embedded fax server that faxes files received from a remote client.

Discussion of the Background:

A conventional fax machine requires a user to leave his or her desk to physically carry a document to be faxed to the fax machine. Since many documents are stored electronically, methods for sending faxes from a personal computer (PC) have been developed. This has eliminated the requirement that users physically carry documents to be faxed to a fax machine, saving both time and money.

Examples of popular PC fax applications are Symantec Corp.'s WINFAX PRO 9.0 and the Computer Associates' BITWARE 4.0. PC fax applications permit documents electronically stored on a PC to be faxed via a modem attached to the PC. However, PC fax applications are for single users and are not a network fax solution. Additionally, PC fax software has to be individually installed on each PC, a modem is required for each PC, and users are required to connect their modem to a conventional public switched telecommunications network (PSTN) and set the modem settings.

Internet or local area network (LAN) fax machines, such as the Ricoh 4800L, can be connected to a LAN. Internet fax machines permit a client PC to send Group 3 (G3) fax messages to the Internet fax machine over the LAN. The Internet fax machine then sends the fax message to a destination such as a G3 fax machine connected to a PSTN or an e-mail address. However, conventional Internet fax machines require PC fax software such as WINFAX, and a COM redirector driver, which redirects fax messages from the PC fax software to an ETHERNET card rather than the COM port. Thus, use of conventional Internet fax machines requires installation of specialized software on each PC.

Windows NT and Unix fax servers have developed rapidly since the T.37 Internet Fax Standards were approved by the ITU-T (International Telecommunications Union-Telecommunications Sector) in June 1998. The T.37 recommendations define the service, file formats and addressing methods for the "Simple Mode" of Internet Fax via e-mail as RFCs (Request for Comments) 2301-2305. The Fax Server is usually directly integrated with the e-mail server and works as a gateway between Internet e-mail and G3 facsimile. The fax server retrieves the fax messages from the message queue of the e-mail server and sends the attachments as well as text messages to the destination via the PC fax modems. Some vendors (e.g., Biscom Corporation) have also developed Web page user interfaces for their fax servers. The fax server allows all network users to send and receive faxes right from their desktop. However, an extra workstation (e.g., NT or Unix workstation) is required to run the fax server. The addressing methods for Internet Fax is not intuitive and requires training in order to use Internet fax services. For example, the Internet fax message addressing syntax "Fax=/num=1408-954-5353/name=John@faxserver.com" indicates that a fax is to be sent to 1408-954-5353. The intended receiver is John, and "faxserver.com" is the domain name of the fax server. Some fax servers are inconvenient to use because they require fax client software to be installed on each user's PC before he can use the fax services. The fax servers do not provide any means for a user to directly fax a hardcopy of a document to a fax number as a normal G3 fax machine does.

Printer servers have been implemented to allow administrative tasks to be remotely executed via a hyper text transfer protocol (HTTP) server resident on a printer. For example, Hewlett Packard's HP LaserJet 8100 series printers include HTTP servers. However, such HTTP servers are merely provided so that a user can access configuration and diagnosis information (e.g., network status, device identification, system configuration, security, diagnosis, and technical support) from a remote client connected to the HTTP server. Such HTTP servers do not provide direct printing services via a Web browser running on a client connected to the HTTP server.

SUMMARY OF THE INVENTION

Accordingly, one object of the present invention is to provide a novel network fax machine that includes a hyper text transfer protocol (HTTP) server.

Another object of the present invention is to provide a fax server, method, and software for providing a fax service through a Web page user interface.

It is yet another object of the present invention to provide a network fax machine having an HTTP server with a uniform resource locator (URL) that enables users to immediately access the network fax machine over a network such as the Internet, upon entering the URL at a client workstation.

It is an even further object of the present invention to provide a fax server, method, and software for sending fax messages through a network fax machine that eliminates the need for installing PC fax software and PC fax hardware, such as a fax modem.

It is still yet another object of the present invention to provide a fax server, method, and software which enables computers installed on a network to immediately access a network fax machine as soon as the network fax machine is installed on the network because the computers do not have to be altered.

These and other objects are achieved according to the present invention by providing a novel network fax machine, method, and software for faxing files received from a remote client. The network fax machine includes a server unit and a fax control unit. The server unit provides an HTML document form to a remote client and receives a fax request from the remote client. The fax request includes an identifier corresponding to a destination fax machine and zero or more attached files to be faxed to the destination fax machine. The fax control unit is configured to use the identifier to connect the network fax machine to the destination fax machine and is configured to send the file to the destination fax machine by facsimile communication. Since fax document forms are sent from the network fax machine to the client, there is no need to install fax software on the client. Further, there is no need to install fax hardware, such as a fax modem, if the client and server are connected by a network.

Preferably, the server unit is an HTTP server configured to serve a hypertext markup language (HTML) document to the remote client. In this case, the HTML document can include the fax document form sent to the remote client. Accordingly, the present invention provides a fax service through a Web page user interface. Moreover, the HTTP server has a URL, which permits users to immediately access the network fax machine over a network such as the Internet.

In a preferred embodiment, the network fax machine includes a network interface card programmed to provide the HTTP server. Additionally, the network interface card can be programmed to provide a common gateway interface (CGI) application and a fax job manager. The CGI application is configured to read and parse the fax information received by the server unit, and the fax job manager is configured to convert the file to be faxed into a facsimile format.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete appreciation of the invention and many of the attendant advantages thereof will be readily obtained as the same becomes better understood by reference to the following detailed description when considered in connection with the accompanying drawings, wherein:

Figure 1 is a schematic illustration showing a Web fax server embedded in a G3 fax machine according to the present invention and in communication with a client or a PC running Web browser software;

Figure 2 is a schematic illustration of an exemplary network on which the Web fax server of Figure 1 can be implemented;

Figure 3 is a schematic illustration of the internal components of the Web fax server of Figures 1 and 2 and of the interrelationship between the Web client, a public switched telecommunications network (PSTN), a normal G3 fax machine, and the internal components of the Web fax server;

Figure 4 is a graphical user interface (GUI) for providing a fax document form as a Web page served to the Web client of Figures 1, 2, and 3 by the HTTP server of Figure 3;

Figure 5 is a GUI providing a fax job log as a Web page served to the Web client of Figures 1, 2, and 3, by the HTTP server of Figure 3;

Figure 6 is a flow chart explaining how fax requests are made and confirmed using the Web browser of Figure 1 and the HTTP server of Figure 3;

Figure 7 is a flow chart explaining the operation of the common gateway interface (CGI) application used in the Web fax server of Figures 1, 2, and 3;

Figure 8 is a flow chart explaining the operation of the parse client data module of the CGI application shown in Figure 3;

Figure 9 is a flow chart explaining the operation of the fax job manager used in the Web fax server of Figures 1, 2, and 3; and

Figure 10 is a schematic diagram of a general purpose computer system that can be programmed to perform the special purpose function(s) of one or more of the devices shown in Figures 1, 2, and 3.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring now to the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views, and more particularly to Figure 1 thereof, there is shown a Web fax server 102, a Web client 104, and a expanded screen shot 106 provided by a graphical user interface (GUI) of the Web client 104, using Web browser software. The terms "Web fax server" and "network fax machine" are used interchangeably to describe a fax machine that includes an embedded fax server.

The Web fax server 102 and the Web client 104 are connected and communicate in a client/server relationship. Thus, the Web fax server 102 and the Web client 104 can advantageously make use of distributed intelligence and processing. The Web fax server 102 performs all of the functions of a traditional fax machine. These traditional functions include scanning images from documents, converting the scanned images into digital data, sending the digital data to a fax machine over a network, receiving digital data from a fax machine, converting the digital data to an image, and printing the image on paper.

As shown in Figure 1, the Web fax server 102 and the Web client 104 communicate using hypertext transfer protocol (HTTP); however, the Web fax server 102 and the Web client 104 may communicate using any suitable network protocol language or application level protocol for distributed systems. Preferably, the protocol used by the Web fax server 102 and the Web client 104 permits transfer of hyper medium information.

Accordingly, the present invention can be implemented on the World Wide Web or any other suitable network or medium for transferring files between a server and a client connected to a network. The design and implementation of various methods of database networking and Internet communications are described in Liu *et al.*, "Managing Internet Information Services," O'Reilly & Associates, Inc., 1994; Comer, "Internet Working with TCP/IP Volume I: Principles, Protocols, and Architecture," 2nd ed., Prentice-Hall, Inc., 1991;

Comer and Stevens, "Internet Working with TCP/IP Volume II: Design, Implementation, and Internals," Prentice-Hall, Inc., 1991; Comer and Stevens, "Internet Working with TCP/IP Vol. III: Client-Server Programming and Applications," Prentice-Hall, Inc., 1993; Khoshafian *et al.*, "A Guide to Developing Client/Server SQL Applications," Morgan Kaufmann Publishers, Inc.; Hamilton *et al.*, "JDBC Database Access with Java, A Tutorial and Annotated Reference," Addison-Wesley Pub. Co., 1997; and Francis *et al.*, "Professional Active Server Pages 2.0," Wrox Press Ltd., 1998; each of which is incorporated by reference herein.

The Web client 104 is a personal computer (PC), mini-computer, workstation, or other computer or processor for sending information to the Web fax server 102 and for receiving files and other information from the Web fax server 102.

The display screen 106 is generated with a Web browser application or software running on the Web client 104. The Web browser application causes Web pages (or hypertext markup language (HTML) pages) downloaded from the Web fax server 102 to be displayed on the screen of the Web client 104.

Figure 2 is an exemplary schematic diagram showing how the Web fax server 102 is integrated within an intranet 202. The intranet includes the Web client 104, a Web client 208, a Web client 210, a mail server 204, the Web fax server 102, and a proxy server 212.

The Web clients 208 and 210 are any processing device (e.g., a mini-computer, PC, or workstation) and can be the same or different from the Web client 104. As shown in Figure 2, the intranet 202 also includes a mail server 204 and a proxy server 212 for interfacing the intranet 202 with the Internet 213. The Web client 104, the Web client 208, the Web client 210, the mail server 204, the proxy server 212, and the Web fax server 102 are all in communication with one another over the intranet 202. The Web fax server 102 is connected to a public-switched telephone network (PSTN) 214 and/or any other network or connection over which fax messages are transmitted.

Fax machines 216 and 218 are connected to the PSTN 214. The fax machines 216 and 218 are conventional fax machines or alternatively configured as Web fax servers, such as the Web fax server 102. The fax machines 216 and 218 communicate with each other and the Web fax server 102, using Group 3 (G3) protocols or any other suitable protocol.

Figure 3 is an exemplary schematic diagram of the Web fax server 102 and the connections between the Web client 104, the Web fax server 102, the PSTN 214, and the fax machine 216.

As noted above, the Web client 104 is a workstation, PC, computer, mini-computer, or other processing device that functions as a client in a client/server architecture arrangement. Web browser software or applications are running on the Web client 104 and provide a GUI for the user to enter fax requests. A fax request includes information such as the name of the intended recipient of the fax, the fax number of the destination fax, the e-mail address of the sender, the subject of the fax, a message, and zero or more attached files to be faxed to the destination fax machine. If zero files are attached, the user can still send text messages by typing text messages in the text message box of the field 410 (Figure 4). The files to be faxed may be stored on the Web client 104 or remotely from the Web client 104. The Web client 104 preferably communicates with the Web fax machine 102 using HTTP protocol, as noted above.

In a preferred embodiment, the Web fax server 102 is a Ricoh 4800L fax machine that includes a network interface card (NIC) 302 and a fax control unit (FCU) 320.

The NIC card 302 is programmed to function as a Web server. The software used to program the NIC card 302 includes the following four software modules: an HTTP server 304; a common gateway interface (CGI) application 306; a fax job queue 308; and a fax job manager 310. The fax control unit 320 communicates with the fax job manager 310 and controls the delivery of fax messages to the fax machine 216 via the PSTN 214.

The HTTP server 304 supports connections with multiple clients (e.g., the Web client 104) and/or multiple Web browsers. The HTTP server 304 generates and sends a Web fax home page to the Web client 104. The Web fax home page is a fax document form used to collect fax information input by a user via a Web browser GUI. The HTTP server 304 also launches the CGI application 306 when fax information from the Web client 104 arrives at the HTTP server 304. Additionally, the HTTP server 304 sends back the server processing status to the Web client 104.

The CGI application 306 unpacks and processes fax information sent to the HTTP server 304 from the Web client 104. CGI is a known standard for external gateway programs to interface with information from servers. The CGI application could be written in any

software language that can read standard input (STDIN), write to standard output (STDOUT), and read environment variables, for example. Thus, the CGI application could be written in C, C++, PERL, or with shell scripting, for example. The CGI application 306 reads streams of client data (e.g., fax information retrieved via the fax document form) through the standard input and parses the streams of client data. Additionally, the CGI application 306 stores variable values into data structures and writes the fax information (e.g., subject, message, and attached binary files) into temporary files. The attached binary files are the documents that are to be faxed from the web client. The CGI application 306 also enters fax messages, received as streams of client data from the HTTP server 304, into the fax job queue 308.

The fax job queue is an application, program, or a script implemented as a text file, for example. If the fax job queue 308 is a script implemented as a text file, then each fax job to be sent can be represented as a single line of the text file. Thus, each line in the text file corresponds to one fax job and contains at least eight fields. These fields are date, time, receiver's name, sender's email address, temporary subject file path and name, temporary message file path and name, temporary attached file path and name. The temporary files can be stored anywhere. By default, they would be stored in the same directory as the fax job queue file. For example, following is one line of text in the fax job queue file representing one fax job:

11/30/99, 15:01:50, Sean Hou, 954-5353, hou@fax.ussj.ricoh.com,
d:\Httpd\Faxjob\fax00001.sub,
d:\Httpd\Faxjob\fax00001.msg, d:\Httpd\Faxjob\fax00001.doc

The fields are separated by commas. The date is 11/30/99, the time is 15:01:50, and the intended receiver is Sean Hou. The destination fax number is 954-5353. The sender's email address is hou@fax.ussj.ricoh.com. The temporary subject file is d:\Httpd\Faxjob\fax00001.sub. The temporary message file is d:\Httpd\Faxjob\fax00001.msg. The temporary attached file is d:\Httpd\Faxjob\fax00001.doc. The file extension ".sub" denotes the temporary subject file, which stores the subject the user entered in his fax request. The file extension ".msg" denotes the temporary message file, which stores the text message the user entered. The file extension ".doc" denotes the temporary MS Word file, which is attached to be faxed.

The fax job temporary files are automatically generated by the computer program. The program first tries to detect if fax00001.* exists or not. If yes, the temporary file name to be used by the next fax job is automatically advanced by 1. Thus, the temporary file name for the next fax job would be fax00002.sub, fax00002.msg, or fax00002.doc, for example, assuming fax00002.* does not already exist. If fax00002.* does exist, the next fax job temporary file name would be fax00003 plus the file extension. The fax job temporary files are continuously named in such a way until the last name fax99999 is designated. The fax job temporary files are immediately deleted once the fax job has been executed by the fax job manager 310, so that the same temporary file name will be immediately available for the next fax job to use. The fax job queue 308 operates on a first-in-first-out (FIFO) basis, so that new fax jobs are attached at the end of the fax job queue 308.

The fax job manager 310 includes the following software modules: a simple mail transfer protocol (SMTP) client 312, a file converter 314, a fax job dispatcher 316, and a fax job monitor 318. The fax job dispatcher 316 runs a fax job dispatching timer with a time interval. At the end of each time interval, the fax job dispatcher 316 reads the first fax job from the fax job queue 308. For example, if the time interval of the fax job dispatching timer is every five seconds, then every five seconds, the fax job dispatcher 316 reads the first fax job from the fax job queue 308.

The file converter 316 is a file conversion engine for converting different types of file formats into fax-ready formats. For example, the file converter 314 receives a Microsoft WORD file and then converts it to a Group 3 TIFF format file. The fax job dispatcher 316 uses the file converter 314 to translate an attached file into fax-ready format. Thus, a user may select any type of file to fax, for example, Microsoft WORD files, POWERPOINT files, EXCEL files, Adobe ACROBAT files, etc. File extensions are used by the file converter 314 to determine the type of file. After conversion, the fax job dispatcher 316 dispatches the fax job to the fax control unit 320.

The fax job monitor 318 keeps a log of fax job activities of the Web fax server 102 and generates for display a list of fax jobs that have been dispatched. The list of fax jobs are delivered to the Web client 104 in the form of a Web page, such as the fax job log 500 (Figure 5). The fax job log 500 is preferably delivered to the user at the Web client 104 at the request of the user.

This invention includes the computer screen interface and the associated program used to generate the interface (e.g., a GUI), which is used for interaction with people (i.e., users) who are associated with and carry out the operation of the invention. For example, the inputs of the invention are entered through the user interface of the screen, and the outputs are displayed on the screen and/or generated on printed paper.

Figure 4 is a GUI implemented as a Web fax home page 400. The Web fax home page 400 is displayed by a Web browser, such as NETSCAPE, running on a client (e.g., the Web client 104). The Web fax server home page 400 includes a field 402, a field 404, a field 406, a field 408, a field 410, a field 412, a field 414, a field 416, and a field 418. The field 402 is for inputting the name of the receiver of the fax message. The field 404 is for inputting the telephone number of the fax machine of the receiver (i.e., the destination fax machine 216). The field 406 is for inputting the e-mail address of the sender. The field 408 is for inputting the subject of the fax message. The field 410 includes a text message box for inputting a message and scroll bars for scrolling through messages that do not fit entirely within the message field. The field 412 is a button, which allows a user to browse a storage device connected to the Web client 104 for files to be sent as fax messages. Such storage devices may include hard disk drives, floppy disk drives, network databases, or any other suitable medium for storing files to be sent by facsimile communication to the fax machine 216. The field 414 is for manually inputting a file to be faxed. The field 416 clears the inputs contained in the fields 402, 404, 406, 408, 410 and 414. The field 418 is a button, which when selected, causes the fax message, including fax information to be sent to the HTTP server 304 from the Web client 104.

Figure 5 shows a fax job log 500 sent to the Web client 104 from the HTTP server 304 in the form of a Web page. The fax job log 500 includes a field 502, a field 504, a field 506, a field 508, a field 510, a field 512, a field 514, a field 516, and a field 518, which are arranged in a single record. The field 502 displays the fax job number. The field 504 displays the status of the fax job (e.g., whether the fax message was sent successfully by the fax control unit 320 or whether an error occurred). The field 506 displays the date that the fax was sent by the fax control unit 320. The field 508 displays the time that the fax message was sent by the fax control unit 320. The field 510 displays the name of the intended receiver of the fax message. The field 512 displays the fax number of the destination fax machine

216. The field 514 displays the e-mail address of the sender. The field 516 displays the fax job ID. The fax job ID is used in generating the temporary fax job files for a particular fax job, which is useful for software debugging. The field 518 displays comments, if any, corresponding to each fax job number stored in the field 502. Thus, as shown in Figure 5, fax job number 0005 was sent September 28, 1999, at 17:31:30 to John at fax number 434-4390. The sender's e-mail was hou@fax.ussj.ricoh.com. As shown in field 504, an error occurred, and the fax job ID displayed in field 516 is fax00001. In the field 518 the error message "ERR=01" is displayed, indicating that the error code for the error is 01. The error code 01 corresponds to a particular type of error identified in a manual or on an on line help screen, for example.

Figure 6 is a flowchart showing how the Web client 104 and the HTTP server 304 operate in parallel to provide a Web fax service. Steps 602, 604, 606, 608, 610, and 612 are performed on the Web client 104, which is preferably an HTTP client running Web browser software that displays Web pages served by the Web fax server 102. Steps 614, 616, 618, 620, 621, and 622 are performed by the HTTP server 304.

First, the processing of the Web client 104 is described. In step 602 a user at the Web client 104 enters the URL or IP Address corresponding to the Web fax server 102. Once the Enter button is pressed in step 602, the Web client 104 will attempt to connect to the HTTP server 304 of the Web fax server 102. Once the Web client 104 connects to the HTTP server 304, the Web client 104 displays a fax document form (e.g., the Web fax home page 400) received from the HTTP server 304. Then, in step 606 the user enters information into some or all of the fields 402, 404, 406, 408, 410, and 414 displayed on the Web fax home page 400.

The user sends the information input in step 606 to the HTTP server by selecting (i.e., clicking on) the button in field 418 of the Web fax home page 400. After the user input data is received and processed by the HTTP server 304, in step 610 the Web client 104 displays the response of the HTTP server 304. Such responses include an indication whether the fax request was successfully received. The user will receive an e-mail reporting his fax status if he has entered his e-mail address into the field 406. In step 612, the user is prompted by the Web client 104 whether there are additional fax messages to be sent. Such prompting can be performed with a Web page sent to the Web client 104 from the HTTP server 304. If the user

has finished sending fax messages, then the process ends. If the user indicates that additional fax messages are to be sent, then the process returns to step 604.

Next, the processing of the HTTP server 304 is described with reference to steps 614, 618, 620, 621, and 622. In step 614, the HTTP server 304 is idle and waits for a connection. Upon receiving a request for a connection from the Web client 104, the HTTP server 304 connects to the Web client 104 and sends the Web fax home page 400 to the Web client 104. In step 620 the HTTP server 304 receives user input data including information such as the receiver's name, the receiver's fax number, the subject of the fax, and/or any other information input by the user into the Web fax home page 400 in step 606. After receiving the user input data in step 620, the HTTP server 304 launches the CGI application 306 in step 621. The CGI application 306 is described in further detail below with reference to Figure 7. In step 622, the HTTP server 304 generates the server response and sends the server response to the Web client 104 in the form of a Web page. Then, the process returns to step 614.

Figure 7 is a flowchart explaining the operation of the CGI application 306. The CGI application 306 is programmed and implemented using any suitable compiler, such as a C ++ compiler. In step 702, the CGI application 306 determines the size of the client data in bytes. The size of the client data in bytes is represented by the environment variable CONTENT LENGTH. If a C ++ compiler is used, the original value of content length is a character string obtained through the standard library call getenv(). Then, the string value of CONTENT_LENGTH is converted to an unsigned long integer using the standard library call atoul().

In step 704, the CGI application 306 compares the integer value of CONTENT_ LENGTH to zero. If the integer value of CONTENT_LENGTH (i.e., the client data size in bytes) equals zero, then a "no data error" message is reported to the HTTP server 304 in step 706. If the integer value of CONTENT_LENGTH is greater than zero, then in step 708 the CGI application 306 allocates a memory buffer of a size at least as great as the client data size. Then, in step 710 the CGI application 306 determines whether the allocation of the memory buffer was successful. If the allocation of the memory buffer is unsuccessful, then in step 712 the CGI application 306 generates a "memory allocation error" message, which is sent to the HTTP server 304. If the allocation of the memory buffer is successful, then in step 714 the CGI application 306 calls a low level I/O routine to read the client data from the

standard input. Once the client data is read into the memory buffer, the memory buffer is passed as a parameter to a parse client data module, which is described in further detail below with reference to Figure 8. If the CGI application 306 is unsuccessful in parsing the client data, then in step 720 a "parse client data error" message is generated and sent to the HTTP server 304. If the CGI application 306 successfully parses the client data, then in step 724 the CGI application generates a server response, which is preferably in HTML. As noted above, in step 622 (Figure 6) the HTTP server 304 sends the server response to the user to the Web client 104, which displays the server response in step 610.

Figure 8 is a flowchart showing how the parse client data module 307 of the CGI application 306 parses client data. The present example assumes that client data is received in a stream separated by "name=" substrings. In step 802, parse client data module 307 finds the next "name=" substring in the client data stream. In step 804, the parse client data module 307 determines whether a "name=" substring was found in the client data stream. If a "name=" substring is not found in the client data, then the process ends. If a "name=" substring is found in the client data stream, then in step 806 the parse client data module 307 retrieves the value of the variable name. Then, in step 808 the parse client data module 307 identifies the value corresponding to the variable name in the client data stream and stores the variable value in any suitable data structure. In step 810, the parse client data module 307 determines whether the variable value of the name is a file name. If the variable value is not a file name, then in step 812 the parse client data module 307 saves the variable value in a data structure, and the process returns to step 802. If the parse client data module 307 determines that the variable value retrieved in step 808 is a file name, then in step 814 the parse client data module 307 generates a fax job file name. Then, in step 816 the parse client data module 307 unpacks the file associated with the file name and stores the client data into temporary fax job files. Then, in step 818 the parse client data module 307 enters the user's fax request into the fax job queue. Thus, the parse client data module 307 opens the fax job queue file and appends a new line of text in that file.

Figure 9 is a flowchart showing the processing performed by the fax job manager 310 after one or more fax jobs are entered into the fax job queue 308 in step 818. In step 902, the fax job dispatcher 316 reads the fax job queue file and retrieves the first fax job from the fax job queue 308. Then, in step 904 the fax job manager 310 determines whether the fax job

queue 308 is empty. If the fax job queue 308 is not empty, in step 908 the file converter 314 decodes the fax job in step 906 and converts the attached file to a fax ready format. Then, in step 910, the fax job dispatcher 316 dispatches the fax job to the fax control unit 320. Additionally, the SMTP client 312 sends back the fax job processing status (e.g., "successfully sent," "error," etc.) to the e-mail address (in the field 406) of the user of the Web client 104. In step 912 the fax job monitor logs the fax job log 500.

Then, in step 914, the fax job dispatcher 316 determines whether the predetermined time interval has expired. If the predetermined time interval has not expired, then in step 916 the fax job dispatcher 316 remains idle. If the predetermined time interval has expired then the process returns to step 902 and the fax job dispatcher 316 retrieves the first fax job from the fax job queue 308. Referring back to step 904, if the fax job dispatcher 316 determines that the fax job queue 308 is empty, then the process proceeds to step 914 and the fax job dispatcher 316 remains idle until the next predetermined time interval has expired.

All or a portion of the invention may be conveniently implemented using conventional general purpose computers or microprocessors programmed according to the teachings of the present invention, as will be apparent to those skilled in the computer art. Appropriate software can be readily prepared by programmers of ordinary skill based on the teachings of the present disclosure, as will be apparent to those skilled in the software art.

Figure 10 is a schematic illustration of a computer system 1000 for implementing the method of the present invention. The computer system 1000 includes a computer housing 1002 for housing a mother board 1004, which contains a CPU 1006, a memory 1008 (e.g., random access memory (RAM) dynamic RAM (DRAM), static RAM (SRAM), synchronous DRAM (SDRAM), flash RAM, read-only memory (ROM), programmable ROM (PROM), erasable PROM (EPROM), and electrically erasable PROM (EEPROM)), and other optional special purpose logic devices (e.g., application specific integrated circuits (ASICs)) or configurable logic devices (e.g., generic array of logic (GAL) or reprogrammable field programmable gate arrays (FPGAs)). The computer system 1000 also includes plural input devices, such as a keyboard 1022 and a mouse 1024, and a display card 1010 for controlling a monitor 1020. In addition, the computer system 1000 further includes a floppy disk drive 1014; other removable media devices (e.g., a compact disc 1019, a tape, and a removable magneto-optical media); and a hard disk 1012, or other fixed, high density media drives,

connected using an appropriate device bus (e.g., a small computer system interface (SCSI) bus, and enhanced integrated device electronics (IDE) bus, or an ultra-direct memory access (DMA) bus). The computer system 1000 may additionally include a compact disc reader 1018, a compact disc reader-writer unit, or a compact disc juke box, each of which may be connected to the same device bus or another device bus. Although the compact disc 1019 is shown in a CD caddy, the compact disc 1019 can be inserted directly into CD-ROM drives which do not require caddies. In addition, a printer may provide printed listings of the information shown in Figures 4 and 5 or any other data stored and/or generated by the computer system 1000.

As stated above, the system includes at least one computer readable medium or memory programmed according to the teachings of the invention and for containing data structures, tables, records, or other data described herein. Examples of computer readable media are compact discs, hard disks, floppy disks, tape, magneto-optical disks, PROMs (EPROM, EEPROM, Flash EPROM), DRAM, SRAM, SDRAM, etc. Stored on any one or on a combination of computer readable media, the present invention includes software for controlling both the hardware of the computer 1000 and for enabling the computer 1000 to interact with a human user (e.g., a consumer). Such software may include, but is not limited to, device drivers, operating systems and user applications, such as development tools. Such computer readable media further includes the computer program product of the present invention for performing all or a portion (if processing is distributed) of the processing performed in implementing the invention. The computer code devices of the present invention can be any interpreted or executable code mechanism, including but not limited to scripts, interpreters, dynamic link libraries, Java classes, and complete executable programs. Moreover, parts of the processing of the present invention may be distributed for better performance, reliability, and/or cost. Preferably the computer system 1000 runs a browser application for displaying HTML documents provided by the Web fax server 102.

The invention may also be implemented by the preparation of application specific integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art.

Obviously, numerous modifications and variations of the present invention are possible in light of the above teachings. It is therefore to be understood that within the scope

of the appended claims, the invention may be practiced otherwise than as specifically described herein.